|  |  |
| --- | --- |
| Министерство науки и высшего образования Российской Федерации  Санкт-Петербургский политехнический университет Петра Великого  Институт компьютерных наук и технологий  Высшая школа киберфизических систем и управления | |
| ОТЧЁТ  по дисциплине «Теория и технология программирования»  по лабораторной работе №3(8) | |
| **Выполнил**: |  |
| студент гр. 3530902/10003 | Сурьянинова В. В. |
|  |  |
|  |  |
| **Проверил:**  ассистент |  |
|  | Журавская А. |
|  |  |
|  | |

1. Задание

Составить программу для поиска по хешам данных. Хеширование проводить в соответствии с индивидуальными заданиями. В модуле поиска предусмотреть реализацию обработки случая, при котором хэш-коды различных данных совпадают.

Вариант №25: Дана таблица текстовой базы данных записями: фамилия; имя; отчество; телефон (телефонный номер представлен в международной классификации номеров: +7812-123-45-78). Произвести хеширование и поиск по номеру телефона.

1. Текст программы

#include <iostream>

#include <fstream>

#include <string>

**using** **namespace** std;

**bool** enter(string& k){

**bool** f = **true**;

**while** (f == **true**) {

cout << "Enter the phone number: " << endl;

cin >> k;

**if** (k.length() == 15 ) f = **false**;

}

**return** **true**;

}

**int** pow (**int** n, **int** k){

**for** (**int** i = 2; i < k; i ++){

n \*= n;

}

**if** (k == 0) n = 1;

**if** (k == 1) n = n;

**return** n;

}

**int** my\_hash (string s){

**int** t = 0;

**int** p = 31;

**for** (**int** i = 0; i < s.length(); i ++){

t += s[i] \* pow(p, i);

}

**return** t;

}

**int** main() {

string line;

ifstream in("/Users/ehet/Desktop/xcode projects/ttplab3/ttplab3/text.txt");

**int** n = 7, t = 0, count = 0;

**int** hash\_k;

string str0, str2, str3, str1, k;

string str[n], string[n];

**int** hash\_num[n];

**if** (in.is\_open())

{

**while** (getline(in, line))

{

string[t] = line;

str[t] = line.substr(57, 15);

t++;

}

}

in.close();

**for** (**int** i = 0; i < n; i ++){

hash\_num[i] = my\_hash(str[i]);

}

cout << "You should enter the phone number in format: +XXXX-XXX-XX-XX" << endl;

enter(k);

hash\_k = my\_hash(k);

**for** (**int** i = 0; i < n; i ++){

count++;

**if** (hash\_k == hash\_num[i]){

cout << string[i] << endl;

count = 0;

}

}

**if** (count == n) cout << "There's no people with this phone number" << endl;

**return** 0;

}

1. Входные и выходные данные

Входные данные: база данных с ФИО и телефонными номерами в виде txt файла (пример на рисунке 1), номер телефона, введенный с клавиатуры в формате +XXXX-XXX-XX-XX.

Выходные данные: строка с ФИО из базы данных, соответствующая введенному номеру телефона. При отсутствии совпадения по номеру телефона выводится информация о том, что человека с таким номером телефона в базе данных нет.



Рисунок

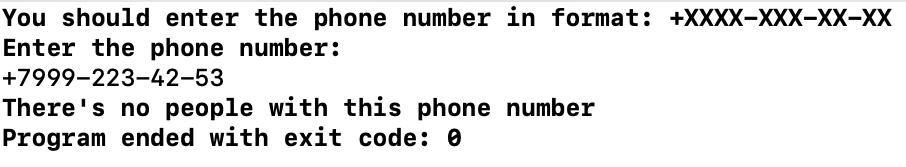
Изображение выглядит как текст

Автоматически созданное описание

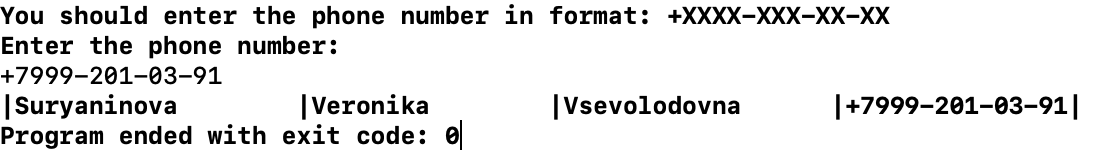
Рисунок - Пример как выглядят захэшированные номера телефонов из базы данных

1. Пример работы программы

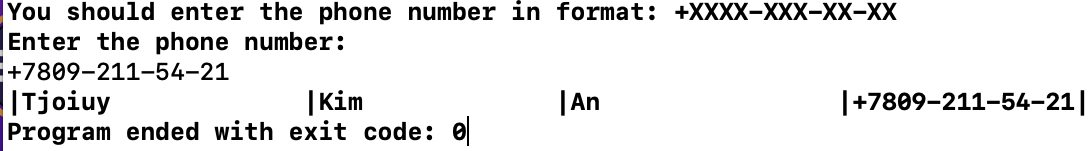
На рисунке 3 представлен пример выполненной программы, где введённый номер телефона не соответствует ни одному телефону из базы данных. На рисунках 4–5 представлен пример, когда программа выводит информацию по номеру телефона. На рисунке представлен случай, когда пользователь вводит неверный формат номера телефона.



Рисунок



Рисунок



Рисунок

Изображение выглядит как текст

Автоматически созданное описание

Рисунок

1. Вывод

В результате проделанной мною работы я написала программу, которая находит в базе данных информацию по человеке по номеру телефона, который к нему привязан. Я использовала хэширование, чтобы зашифровать номер телефона человека, а затем по полученным значениям искать среди них введенный с клавиатуры номер телефона (который предварительно также хэшируется).

Во время выполнения данной лабораторной работы я ознакомилась с основами хеширования на C++, что помогло мне разобраться в различиях между поиском по исходным данным и по зашифрованным.

Эти знания могут помочь мне в написании более сложных программ в будущем.